IBM® Technical Disclosure Bulletin Vol. 36 No. 07 July 1993

Code Generation for an Object Oriented Application

Disclosed with this bulletin are programs which generate the data handling layer of a ProductManager based application. Essential elements of the invention include:

1. A flexible template processing mechanism which operates within the ProductManager environment.

Generation of source code is a special case of the general problem of generating a sequential file which has a predictable pattern but variable content. The predictable pattern is the template for the file, which consists of literal text with substitution variables and procedural exits (macros) to retrieve data and make decisions. The procedural exits are ProductManager metaclasses which inherit from a common class which provides a consistent interface to the template processor. These macros also handle iterations (if lines must be repeated in the output file based on data conditions), and they can invoke other templates to write out lines determined by logic within the macro. These templates can in turn access other macros.

Template creation and editing support is provided. Templates are stored in ProductManager using DB2 sequential files, and they are presented to the user using a text editor available in the ProductManager operating environment (for TSO/MVS, this is the standard ISPF editor).

2. Specific support for code generation. Specific templates and macros for generation of data handling code are provided. These include the following:
   a. Declaration of persistent attributes
   b. The corresponding assign methods
   c. Object oriented clauses to export both the attributes and the assign methods.
   d. The methods needed for communication with panels and Product Data Interface type 2 (PDI2).
   e. The methods needed for object-oriented data base communication deferred from the Persistent Object (EKNDPERS).
   f. Common find and stream methods, with their cursors.
   g. The stream element class.

   The methods for the persistent object are packaged together in a template which provides an independent class specification for all the methods which can be generated. The intention is to develop this portion of the total logic in the class in isolation from other methods which must be hand coded. This template may need slight modifications to fit into a particular application structure.
3. Derivation of metadata from a data model - Tools are provided to strip information about the entities and attributes for an application from a data model, and to convert it into format suitable as input to the code generation process.
4. Support within the Application Model - Code generation is initiated from the Application Model. Some of the relationships required as input for code generation are also specified with the Application Model. These are restricted to ones which are not redundant with the data definitions derived from the data model. The Application Model is automatically updated to reflect the features defined during code generation.

This invention reduces the amount of development effort needed to implement standard data handling code, and improves quality and maintainability by generating this code from standardized templates. It also enhances portability and flexibility. If the data model changes, the data handling code can be regenerated to reflect the change. If a different platform requires a different implementation of data handling, the likelihood is that this can be accommodated by changing the templates. The same holds true for possible performance fixes.

This implementation uses MVS, TSO, DB2 and C. It is not; however, fundamentally dependent on any of these.